09/576,187

WEST Search History

DATE: Tuesday, May 06, 2003

Set Name side by side	Query	Hit Count	Set Name result set
	DB=USPT; PLUR=NO; OP=ADJ		
L26	L25 and (l2 or l6 or l7 or l9 or l13 or l15 or l17 or l4)	3	L26
L25	L24 and l1	5	L25
L24	genealogy	154	L24
L23	l2 and l5 and (l9 or l13 or l15)	1	L23
L22	l2 same l4	0	L22
L21	l2 same l3	0	L21
L20	L18 and (l5 or l8 or l9 or l10 or l11 or l14 or l15 or l16 or l17)	56	L20
L19	L18 and l5 and l8 and (l9 or l10 or l11 or l14 or l15 or l16 or l17)	0	L19
L18	l1 and l2	61	L18
L17	workspace	2725	L17
L16	owner	35724	L16
L15	shared adj1 ownership	32	L15
L14	list near3 owner$1	255	L14
L13	"list of owners"	0	L13
L12	user adj1 intervention	2517	L12
L11	version	147580	L11
L10	copy	100839	L10
L9	ownership	6932	L9
L8	L7 or l6	1437	L8
L7	configuration adj1 object$1	1160	L7
L6	configuration adj1 item$1	281	L6
L5	L4 or l3	112	L5
L4	"object-oriented database management system"	88	L4
L3	ODBMS	32	L3
L2	configuration adj1 management	1168	L2
L1	(707/203 OR 707/200 OR 707/103).CCLS.	1506	L1

END OF SEARCH HISTORY

L20: Entry 2 of 4 File: USPT Mar 31, 1998

DOCUMENT-IDENTIFIER: US 5734899 A
TITLE: Device for managing data in a version

Abstract Text (1):
A version control device comprises an entire-generic information storage object for storing version control information common to data under the application of a version control, version-generic information storage objects which are managed by the entire-generic information storage object and stores version control information corresponding to each piece of the data and common to all versions of the data, and version-specific information storage objects which are managed by one of the version-generic information storage object and stores version control information corresponding to each of the versions of the data and specific to each version.

Detailed Description Text (89):
If these cases should be avoided, a rule specific to the version control function should be established such that the owner of any version of an object should be a user who has created the original version, or that the ownership of the object is transferred to the system when the object is updated by a user other than the creator of the original version. Then, the concept of an owner other than the owner managed by the DB is used for desirable access control.

Current US Original Classification (1):
707/203

Other Reference Publication (7):
Moon Jung Chung and Sangchul Kim, "The Configuration Management for Version Control in an Object-Oriented VHDL Design Environment", IEEE, pp. 258-261, 1991.

L23: Entry 10 of 13 File: USPT Mar 7, 2000

DOCUMENT-IDENTIFIER: US 6035297 A
TITLE: Data management system for concurrent engineering

Brief Summary Text (44):
Our solution to the problems of a highly customable Design Control System which can be tailored to grow from current methodologies is a highly modular arrangement which enables the user to install only those pieces which are necessary, and grow the design control system from a fairly simple implementation all the way to a system capable of controlling a large complex project such as the development of the ES/9000 mainframes. Our methods can be implemented on a variety of platforms (client/server, VM, etc) using almost any high Level language. In addition the centralized control repository can be any type of database, a relational database, an object-oriented database, or a file-based database.

Brief Summary Text (45):
We provide a Design Control Data Management System (DMS) capable of managing a coherent set of data objects in a computer environment. The objects are tracked in a controlled manner which permits all pieces to attain the necessary degree of completion. The system is designed to maintain data integrity while allowing multiple users to acquire shared access and ownership to the data objects. The invention works in a client/server domain and permits data to span multiple computer platforms.

Drawing Description Text (18):
FIGS. 16a thru 16b illustrates the main Data Management Configuration screen.

Detailed Description Text (34):
The owner attempting to send data to a Public Library must possess the update lock. If no lock exists, the sender obtains the lock by default. If another user has the lock and the sender is a surrogate, he can obtain the lock (the system immediately notifies the original owner). If the sender is not a surrogate, the action is halted, until ownership is properly transferred.

Detailed Description Text (69):
If the information corresponds to a Bill of Materials (BOM) and the user wants to set the lock on the entire BOM, a BOM Flag will exist allowing him to specify this action. Regardless of how these fields are filled in, all locks will be swill be set individually so they may be removed individually. A lock does not have to be removed the same way it was set. The user will also specify the type of lock, Move, Overlay, or Update (Ownership). The following definitions exist:

Detailed Description Text (72):
Update (Ownership) Locks are the means by which a designer takes ownership of a piece of data. Update locks are designed to prevent multiple designers from updating the same design component in an uncontrolled way, thus resulting in data corruption or lost information. There are two types of Update locks, permanent and temporary.

Detailed Description Text (73):
A permanent Update lock exist when the designer specifically requests to own a piece of data. This is done through a utility, and the DCS keeps track of this ownership. Other designers may copy and modify the data in their private libraries, but any attempt to promote that data into the public library will fail, unless the designer is a designated surrogate of the owner. The only way these locks are removed are by the owner resigning the lock or a surrogate assuming the ownership of the data, and the

corresponding lock. A temporary Update lock exists to facilitate sharing a piece of data among multiple designers. The user can either request a temporary Update lock in advance (i.e. when he begins editing the data), or he can wait until he initiates the promote into the public library. The DCS will first check to see if anyone has a permanent Update lock, and if so, it will only allow the promotion to continue if the user is a designated surrogate. If nobody has a permanent Update lock, then the DCS will issue a temporary Update lock for the time the data remains "en route" to the final promote destination. Once it arrives safely, the temporary Update lock is removed and the data can be claimed for ownership by someone else. Surrogates are "alternate" owners of data. For example, a project may be arranged such that each piece of design is owned by a primary designer, but also has a backup owner (designer) to take over the design during vacations, emergencies, etc.. In this case, the owner can tell the DCS that the backup designer should be a surrogate, thus giving him the right to take ownership of a design component. The surrogate can either use the locking utility to specifically take ownership prior to making any updates, or he can wait until he initiates a promotion. The DCS will check to see if the design component is currently owned, and if so, check to see if the user is a defined surrogate. If both are true, it will give the user the chance to "take ownership" and allow the promote to continue. The original owner would be notified that his surrogate has taken ownership. FIG. 6 illustrates the lock mechanisms for Update locks.

Detailed Description Text (129):
For each level, the Data Manager can add, modify or delete processes. For each process information is required about the type of machine it can run on, any necessary arguments, the result criteria, disposition instructions for the output, whether it's dependent on another process, and whether it should be deferred. The DCS provides Process Specific Boilerplates which can be used to manage process configurations for an entire project. Necessary and required information for each process can be programmed into the DCS, so when a Data Manager attempts to define that process to his library, some of the fields appear with default data already filled in. He can override any of the data.

Detailed Description Text (203):
The present embodiment provides data control and security through a Lock Manager which offers three types of locks. First, there are Out for Update or Ownership locks which permit a user to check out a data object and modify it without fear of another user making a simultaneous update. Our embodiment also provides a means for transferring ownership of a piece of data from the primary owner to a designated surrogate without the primary owner's intervention. Upon completion of the transfer, the primary owner is automatically notified of the ownership transfer. Additionally, the preferred embodiment provides an environment where multiple users can own the same piece of data at different Library Levels.

Detailed Description Text (204):
In addition to ownership locks, the Lock Manager offers Move and Overlay locks which can be used to prevent data from being moved through the DMS or overlaid by the data at lower Levels. It also interacts with the Aggregation Manager to provide locking of entire an Bill of Materials, and it interacts with the Process Manager to provide an interlocking mechanism for data undergoing Automated Library Processing.

Detailed Description Text (271):
Physical segregation of data is accomplished through the Data Management Configuration Utility described later in this disclosure. Whenever a Data Manager establishes a new Library, the utility requires a default physical location to be entered. All data will reside here, unless further segregation is desired. Additionally, as each new Version, Level and File Type is defined to the Control Repository, the opportunity is presented to add sub-directories to increase any desired degree of PFVL segregation. Consider the following:

Detailed Description Text (278):
Data Management Configuration Utility

Detailed Description Text (279):
Our embodiment permits the Data Manager to configure their Libraries in numerous ways. This is accomplished through the use of a Data Management Configuration Utility (also

known as the Package Manager Utility), which can be invoked as an independent program or launched from a design system framework. The only parameter required is the Package Identifier (Pkg. ID), commonly called the Library Name. All other information is entered into the user screens illustrated in FIG. 16 through FIG. 28.

Detailed Description Text (306):
Next, Step 18924 displays the Main Screen depicted in FIG. 30. The screen consists of Main Menu Bar 18861, Profile Headings 18862, and the editable user area 18863. The Profile Headings section displays the titles of each profile in the order specified by the Data Manager's customized configuration settings. The editable area, 18863, displays one user on each line, with the appropriate "Y/-" tokens to denote their association to each profile. The Data Manager can change any of the "Y/-" tokens, but can't change the names of any users from this screen. Data in this window can be scrolled horizontally and vertically via scroll bars 18864 and 18865 respectively.

Detailed Description Text (324):
Once this process is complete for all profiles and all users, the result is a precedence list containing a mixture of existing entries and new entries to be added to the DMS. There should be no overlapping authorities between any of the entries. A simple filter eliminates all the existing authorities and leaves only those which need to be added to the Control Repository. Additionally, if the program is running the Full Update Mode, a separate list is created with all extraneous records that can be removed from the authority tables in the DMS. Since our preferred embodiment embeds the Authority Profile Editor within the Data Management Configuration Utility, the editor does not update the Control Repository directly. Instead, the list of additions and deletions are passed to the Commit Process described in FIG. 49 which, in turn, updates the tables in the Control Repository. The Data Manager must leave the Authority Profile Editor and select the Commit choice from the Main Menu Bar of the Data Management Configuration Utility. It would be appreciated by one skilled in the art that the Authority Profile Editor can exist as an independent entity and could easily incorporate those routines necessary to update the Control Repository.

Detailed Description Text (423):
Since the Data Manager is responsible for assigning and maintaining authorities, all user interfaces with the Authority Manager are contained within the Data Manager Configuration Utility described in FIGS. 16 thru 19970. The preferred embodiment illustrates two distinct user interfaces. The simplest permits the Data Manager to specify the type and granularity of authority on a user by user basis. A simple data entry screen exists whereby the user is identified, the authorities selected, and the Control Repository updated. A detailed description is presented as part of the Data Management Configuration Utility.

Detailed Description Text (428):
In order to preserve data integrity and safety, our embodiment requires all official authorizations to reside in the Authority Tables of the Control Repository. Although the Data Manager can create and edit the Master Authority List and the profiles using any text editor, our embodiment provides a special authority profile editor to facilitate the task. The editor is actually a user interface which interacts with a special algorithm that is responsible for coordinating all profile and Master Authority List changes with the Control Repository tables. As previously stated, the preferred embodiment describes this algorithm and the user interface as part of the Data Management Configuration Utility in FIGS. 16 thru 19970.

Detailed Description Text (429):
To facilitate the exchange of data ownership, our embodiment incorporates a concept known as Surrogates. Users may define other users as Surrogates to act on their behalf. A Surrogate is capable of setting and resetting locks, including ownership locks, without any intervention from the original owner (known as the Ward). This is an especially useful feature if multiple users are collaborating on a single piece of data. The Lock Manager and Library Manager interact with the Authority Manager during certain tasks where Surrogate checking is required. In any situation where a Surrogate successfully confiscates data from the Ward, the DMS automatically notifies the Ward of the event.

Detailed Description Text (430):

FIG. 33 illustrates the Surrogate Editor which is used to browse and assign Surrogates. In our preferred embodiment, only users may add Surrogates for themselves, or authorized Managers may create Surrogates for any users with whom they associate. Since the entire Surrogate apparatus is driven by control tables, our embodiment permits the users of the embodiment to designate the authorized Managers. They may be Data Managers, Personnel Managers, Technical Leaders, Supervisors, or anyone who has the responsibility for assigning data ownership. The single screen shown in FIG. 33 is used to perform all Surrogate editing functions. The top section contains radio buttons, 49110, which offer three choices. The user may ask the editor:

Detailed Description Text (539):
Out for Update Also known as an Update or Ownership lock, this denotes that a user owns a piece of data. Since Out for Update locks are associated with a Package, Filetype, Version and Level (PFVL), our embodiment permits multiple users to share the ownership of the same piece of data. Multiple users can accomplish this by acquiring Out for Update locks at different Library Entry Levels.

Detailed Description Text (543):
In addition, the Lock Manager also permits users to share data by acting as Surrogates for one another. A Surrogate of a user may acquire ownership of the user's data without requiring intervention on the user's part. If a surrogate takes ownership, notification is automatically sent to the original owner.

Detailed Description Text (558):
Otherwise, the lock exists at another Level and is owned by the current user or another user. In this case, Step 68228 allows the user to Confirm the operation. This gives the user the opportunity to abort the operation in the event of unexpected ownership by another user.